An Empirical Evaluation of Using Large Language Models for Automated Unit Test Generation

Year: 2023 | Citations: 368 | Authors: Max Schäfer, Sarah Nadi, A. Eghbali

---

Abstract

Unit tests play a key role in ensuring the correctness of software. However, manually creating unit tests is a laborious task, motivating the need for automation. Large Language Models (LLMs) have recently been applied to various aspects of software development, including their suggested use for automated generation of unit tests, but while requiring additional training or few-shot learning on examples of existing tests. This paper presents a large-scale empirical evaluation on the effectiveness of LLMs for automated unit test generation without requiring additional training or manual effort. Concretely, we consider an approach where the LLM is provided with prompts that include the signature and implementation of a function under test, along with usage examples extracted from documentation. Furthermore, if a generated test fails, our approach attempts to generate a new test that fixes the problem by re-prompting the model with the failing test and error message. We implement our approach in <sc>TestPilot</sc>, an adaptive LLM-based test generation tool for JavaScript that automatically generates unit tests for the methods in a given project's API. We evaluate <sc>TestPilot</sc> using OpenAI's <italic>gpt3.5-turbo</italic> LLM on 25 npm packages with a total of 1,684 API functions. The generated tests achieve a median statement coverage of 70.2% and branch coverage of 52.8%. In contrast, the state-of-the feedback-directed JavaScript test generation technique, Nessie, achieves only 51.3% statement coverage and 25.6% branch coverage. Furthermore, experiments with excluding parts of the information included in the prompts show that all components contribute towards the generation of effective test suites. We also find that 92.8% of <sc>TestPilot</sc>'s generated tests have <inline-formula><tex-math notation="LaTeX">$\leq$</tex-math><alternatives><mml:math display="inline"><mml:mo>≤</mml:mo></mml:math><inline-graphic xlink:href="schaefer-ieq1-3334955.gif"/></alternatives></inline-formula> 50% similarity with existing tests (as measured by normalized edit distance), with none of them being exact copies. Finally, we run <sc>TestPilot</sc> with two additional LLMs, OpenAI's older <italic>code-cushman-002</italic> LLM and <italic>StarCoder</italic>, an LLM for which the training process is publicly documented. Overall, we observed similar results with the former (68.2% median statement coverage), and somewhat worse results with the latter (54.0% median statement coverage), suggesting that the effectiveness of the approach is influenced by the size and training set of the LLM, but does not fundamentally depend on the specific model.